SYNERGETICS
GET IT RIGHT

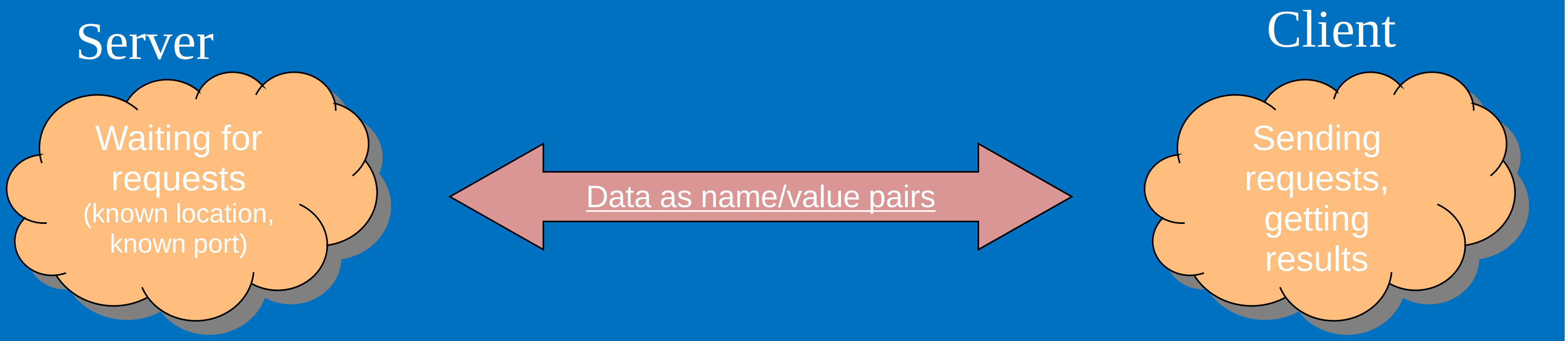
Web Services

Client Server communication

S.N.	Communication	Description
01	HTTP	For web-application. Unidirectional.
02	XML-RPC	A2A, Unidirectional
03	JSON-RPC	A2A, Bidirectional
04	Web Services	A2A, Unidirectional
05	Web Socket	HTTP protocol, Bidirectional.
06	AJAX	For web-application, Unidirectional.

Different ways of communication

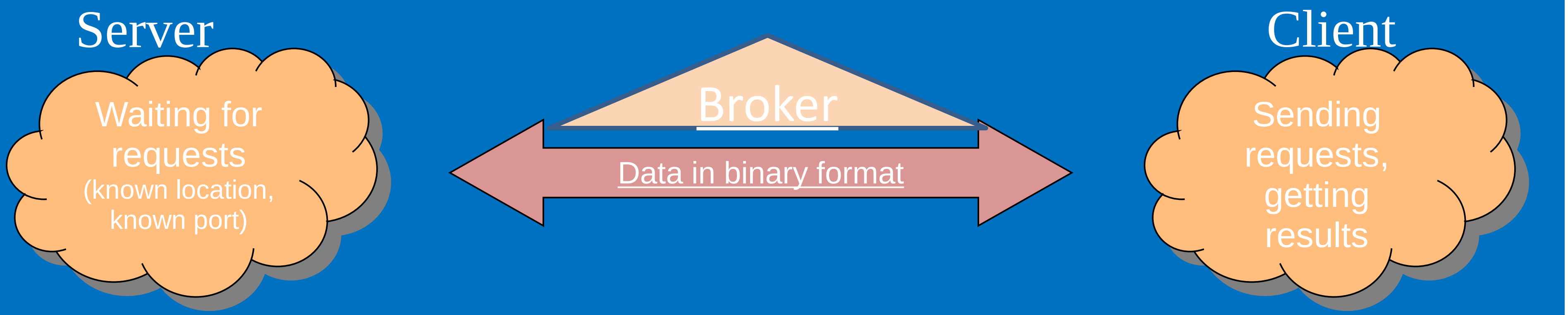
CGI/HTML communication


Server
Waiting for requests
(known location, known port)
Data as name/value pairs
Client
Sending requests, getting results

- ❑ Data transferred as Name-Value pair: Query string or form data;
- ❑ Transport over stateless but standard HTTP protocol;
- ❑ Extra efforts for tracking session. No standards for session management;
- ❑ Server side: Server side script is executed;
- ❑ Limitations for A2A communication.

Different ways of communication

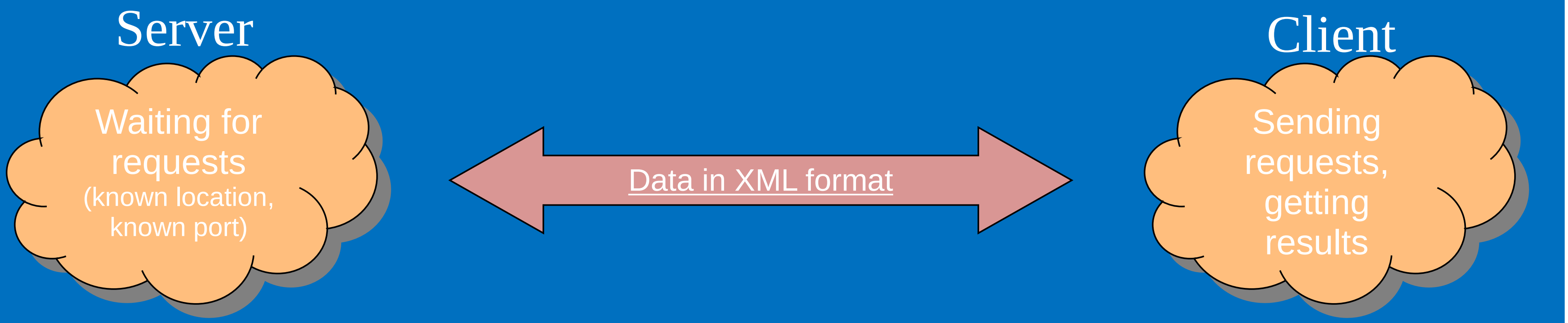
CORBA communication


Server
Waiting for requests
(known location, known port)
Broker
Data in binary format
Client
Sending requests, getting results

- ❑ Data transferred as Object- Not actually but look like;
- ❑ Transport over standard IIOP protocol;
- ❑ User sessions are inter-operable;
- ❑ Server side: RPC calls are made;
- ❑ Designed for A2A and B2B communication;
- ❑ Needs dedicated intermediate level for CORBA server.

Different ways of communication

SOAP based communication


Server
Waiting for requests
(known location, known port)
Data in XML format
Client
Sending requests, getting results

- ❑ Data transferred in well defined XML format;
- ❑ Transport over various standard firewall friendly protocols like HTTP, SMTP etc;
- ❑ User sessions are inter-operable;
- ❑ Server side: RPC calls, Message delivery;
- ❑ Designed for A2A and B2B communication;
- ❑ Lightweight compared to CORBA communication as no need of intermediate level.

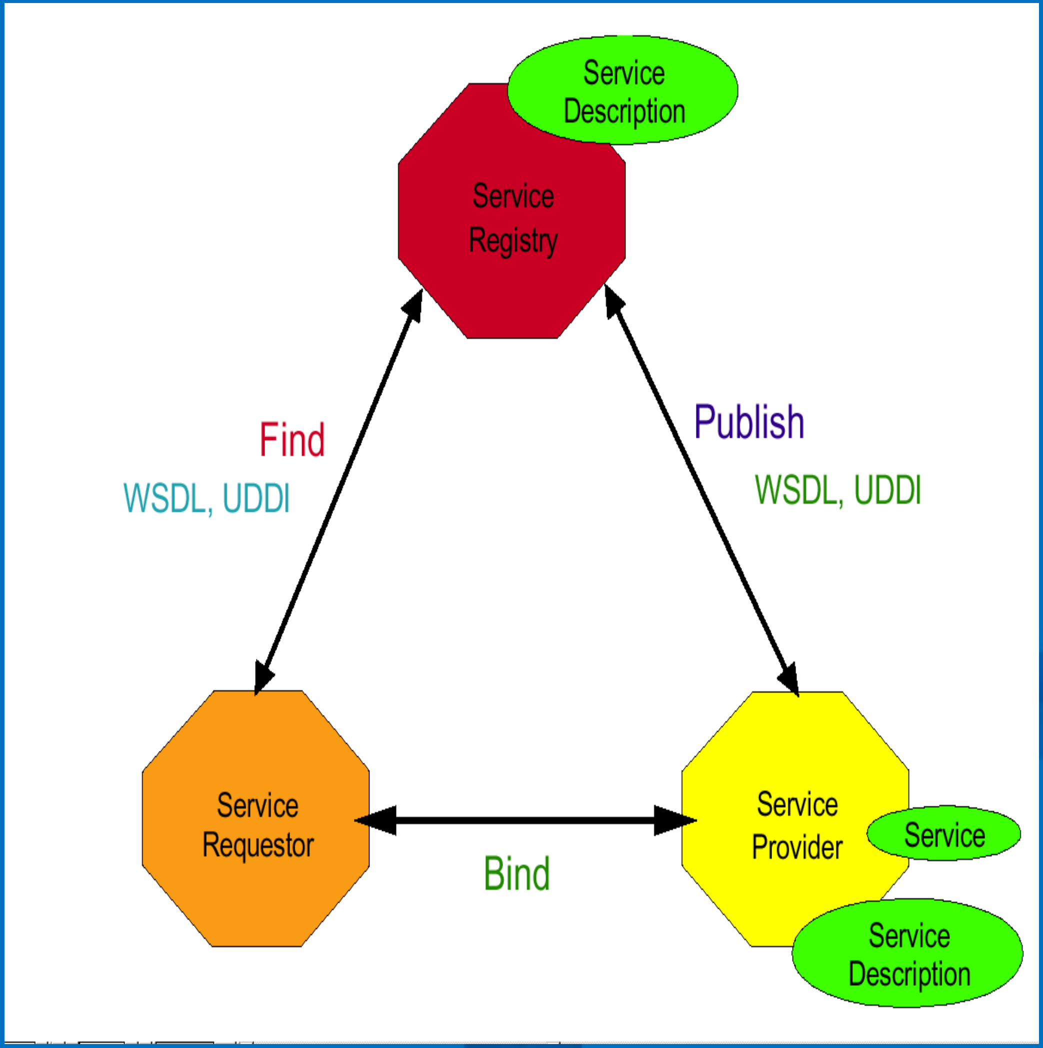
SOAP Based and REST style

- Request/Response Message Exchange Pattern (MEP)
- SOAP is special case of REST Style web service
- SOAP - Simple Object Access Protocol OR Service Oriented Architecture Protocol?
- REST – REpresentational State Transfer is an architectural style in the design of web services.
- REST – Yet to evolve as a standard, an antidote to complexities of SOAP.

Web Services

W3C Consortium

"A Web service is a software application identified by a URI, whose interfaces and bindings are capable of being defined, described and discovered as XML artefacts. A Web service supports direct interactions with other software agents using XML based messages exchanged via internet-based protocols."



Roles in Web Service

Service Provider: Describes the web service on Service Registry through UDDI.

Service Registry: Provides an interface to service requesters to do lookup for services.

Service Requester: Searches a service on Service Registry and get URI for service. Connects to service provider using URI to receive service details in the form of WSDL.

Requester-Provider exchange: WSDL provides details about service. Requester invokes service using description in WSDL.

Web Services

Features

- Open Infrastructure
  - Industry standard- XML, Schema, WSDL, SOAP, UDDI etc.
  - Vendor independent protocols like- HTTP, FTP, SMTP etc.
- Language Transparency
  - Inter-operability across heterogeneous applications.
  - Almost all prominent languages support library
  - Almost all frameworks have built-in support
- Encourage modular design
  - Facilitate integration and layering of services

Web Service standards

XML (eXtensible Mark-up Language)

- It is a de-facto standard for representing hierarchical data.
- Describes the format of the data to be exchanged between two parties.
- Structures data in hierarchical order using tags and elements.
- Allows domain specific customization of tags.

- Other standards associated...
  - XML Namespace: A namespace management standard to identify tags and elements of an xml document uniquely.
  - XML Schema: Provides mean of defining structure, contents and semantics of document.

Web Service standards

SOAP (Simple Object Access Protocol/Service Oriented Archi. Prot.)

- It is a de-facto standard protocol for exchanging XML based messages.
- Exchanges SOAP envelop on HTTP network between two parties.
- SOAP envelop comprises Header and Body.
  - Header holds metadata information about body as well Security and Transaction related information.
  - Body holds actual payload.

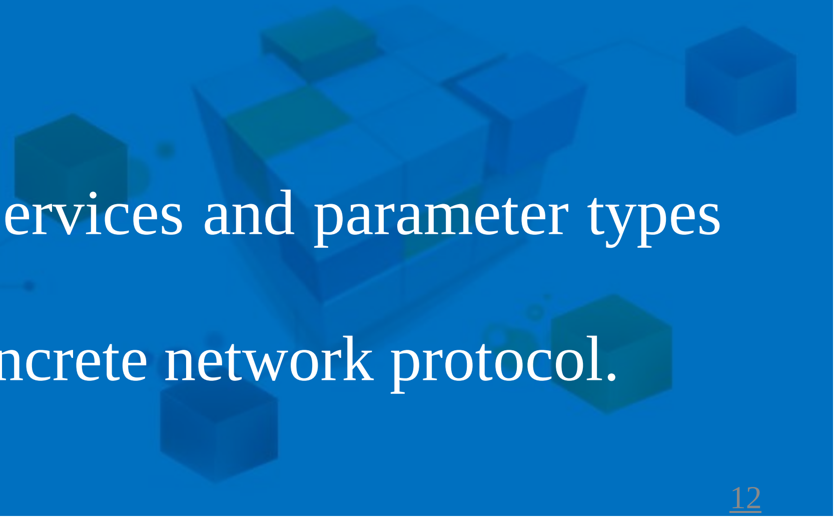
Web Service standards

WSDL (Web Service Description Language)

- A standard to describe web services.
- Web services are described as set of communication endpoints called as ports.

- SOAP envelop comprises Header and Body.
  - Header holds metadata information about body as well Security and Transaction related information.
  - Body holds actual payload.

- Endpoint comprises...
  - Abstract definitions of operations: Services and parameter types services accepting.
  - Binding of abstract definitions to concrete network protocol.



Web Service standards

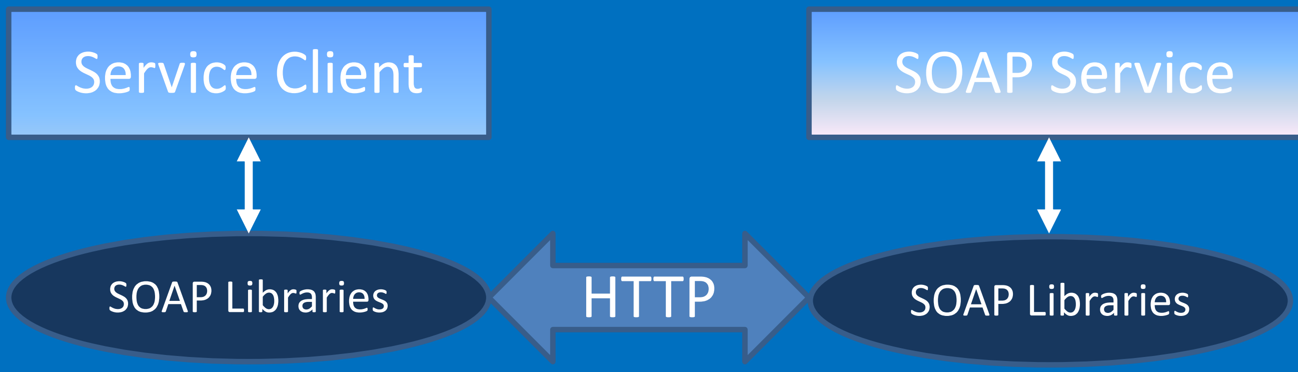
UDDI (Universal Description Discovery Interface)

- It is a de-facto standard mechanism to register and lookup for services.
- Lacks enterprise wide adaption. Enterprises started shipping their own registration interface.

SOAP

- SOAP 1.1 allows HTTP transport while SOASP 1.2 onwards allows HTTP/SMTP etc.
- SOAP 1.1 goes with single document while SOAP 1.2 onwards go with three documents.
- SOAP 1.1 allows XML tags even after SOAP body. For SOAP 1.2, body is the last tag.

Architecture- SOAP based WS



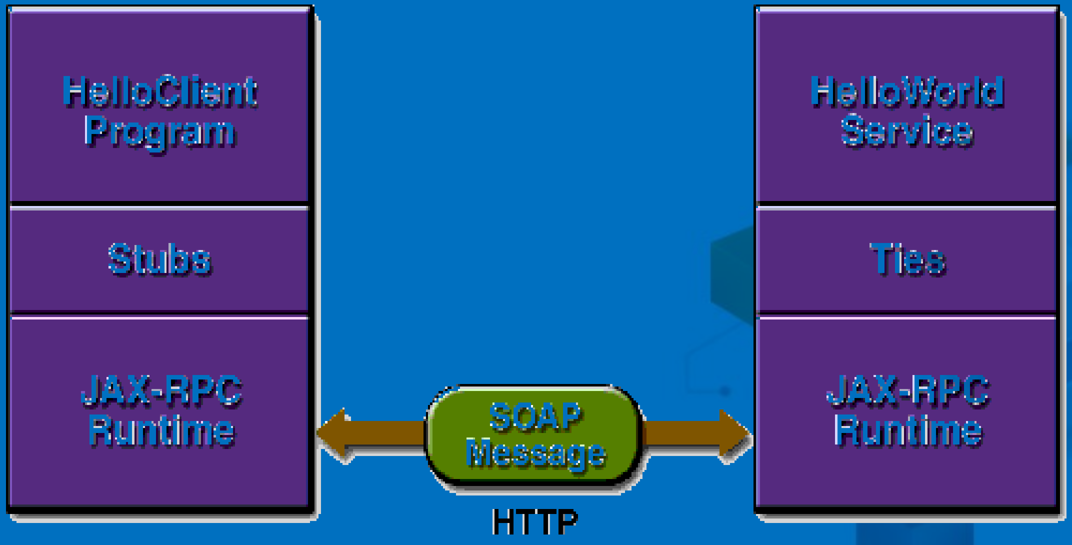
- JAX-WS libraries are shipped with JDK 6 onwards.
- Current version is JAX-WS 2.2.x
- JAX-WS supports SOAP and REST style services.
- JAX-RPC is JAX-WS Ver 1.x
- JAX-WS-ri is Release Implementation of Metro Web Service Stack. It uses annotation thus supported by JDK 5.x onwards.

Document Vs. RPC Style

Document	RPC
SOAP body contains XML document validatable against schema.	SOAP body contains XML representation of method call and primitive data types defined through Holder Classes.
Gives customizable structure of SOAP body.	Structured to represent method name and parameters a method receives.
Literal Encoding: Body contains conform to specific schema.	SOAP Encoding: Message doesn’t conform to specific schema. It defines set of rules based on XML data types to encode the data.
Reference to cyclic object gets repeated.	Useful when passing Cyclic Object Graph.
Message validation by schema.	Can not validate message with XML schema and can not transform message using XSLT.

JAX-RPC

- API for building Web Services and client that use Remote Procedure Calls.
- Calls and Messages are transmitted as SOAP messages over HTTP protocol.
- Big advantage- Platform ubiquitous as uses all standards of W3C like- SOAP, XML and WSDL.



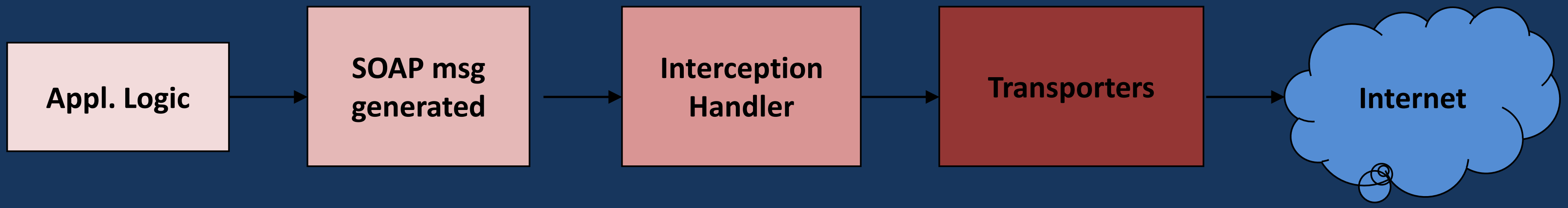
JAX-RPC

- Supported Invocation modes…
  - Synchronous Request-Response
  - One way RPC Mode
  - Non-blocking RPC Mode.

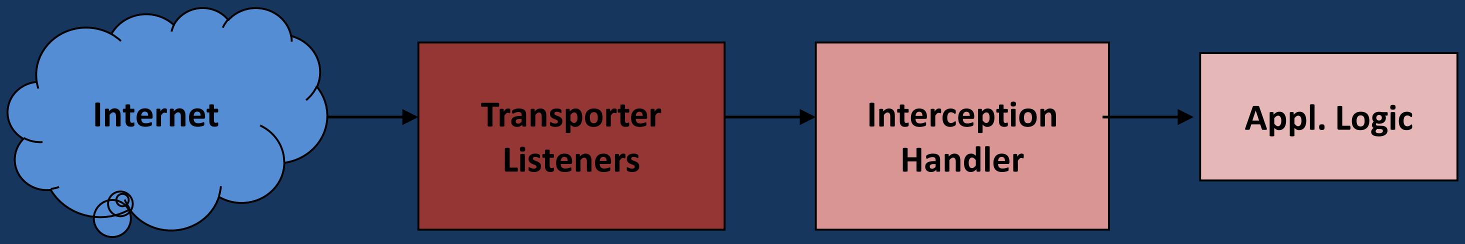
- Parameter Passing Modes…
  - The IN Type: Passed by copy from client to a service. Return value sent in separate object.
  - The OUT Type: Passed without a value from client to a service. Service populates a result in the object and sends back.
  - The INOUT Type: Same object passed by value from client to a service

SOAP message life cycle.

Life cycle...



Life cycle...



SOAP message format

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env=http://www.w3.org/2003/05/soap-envelope  xmlns:wsa="http://www.w3.org/2005/03/addressing">
```

Provides a metadata of message.

```
    <env:Header>
        <wsa:MessageID>http://ws.apache.org/9C21DE32-DB42-1228-C42E-66CB101421AD</wsa:MessageID>
        <wsa:ReplyTo>
            <wsa:Address>http://example.com/projects/clientApp</wsa:Address>
        </wsa:ReplyTo>
        <wsa:Action>http://example.com/axis2/addDocument</wsa:Action>
    </env:Header>
```

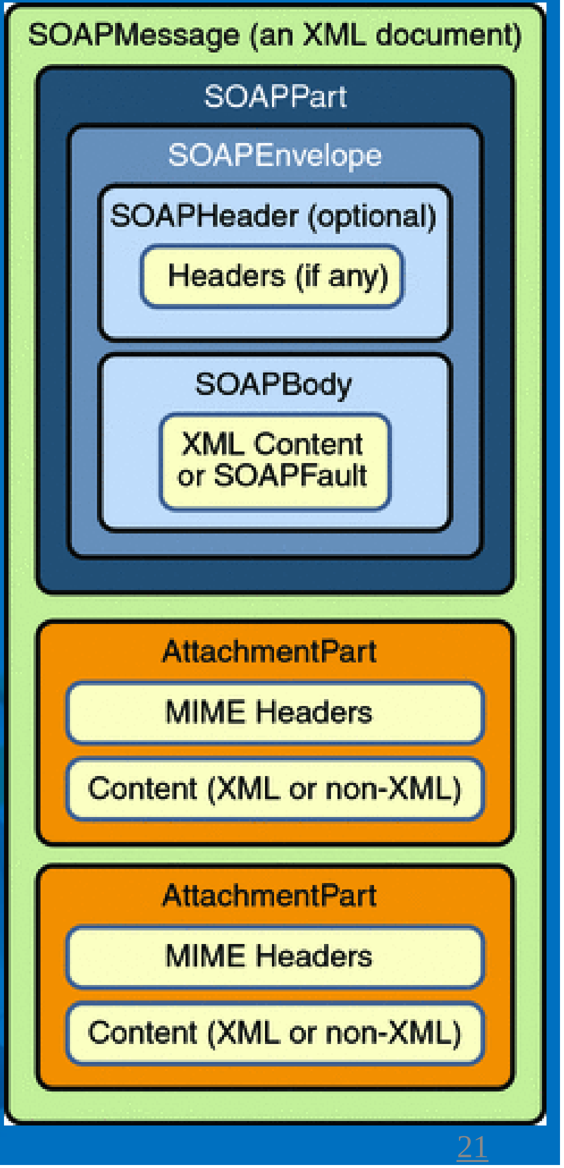
Carries actual payload.

```
    <env:Body>
        <addDocument>
            <docTitle>What I Did On My Summer Vacation</doctitle>
            <docLocation>contentRepos/summerVac.doc</docLocation>
         </addDocument>
    </env:Body>
</env:Envelope>
```

SOAP Message

An ordinary XML document with following elements…

- SOAP Part to hold XML SOAP message.
  - SOAP Envelope: Defines start and end of the message.
  - Header: Optional to hold metadata.
  - Body: Actual XML message payload.
  - Fault: Optional. Holds error if any.
- Attachment Part to hold non-xml data.



JAX P and JAX B

JAXP:

- A Java API for XML parsing. Leverages parsing standards- SAX and DOM parsing.
- Pluggable, Vender neutral, Simple API.
- javax.xml.parsers.SAXParserFactory for SAX parser and javax.xml.parsers.DocumentBuilderFactory for DOM parser.

JAXB:

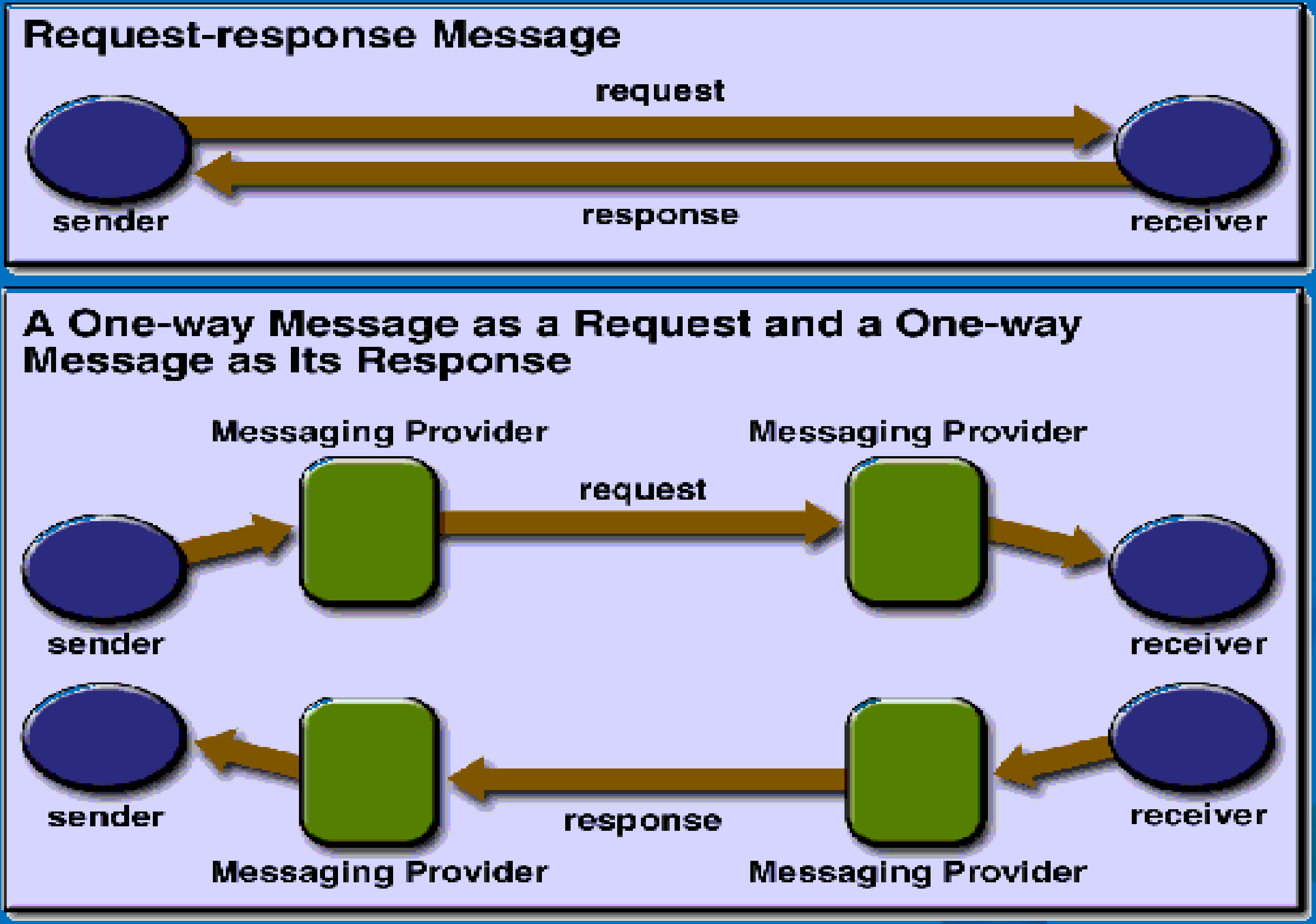
- @XmlRootElement
- @XmlElement
- @ XmlAttibute
- @WebParam

JAXM

- Java API for sending XML document on the net.
- Based on SOAP 1.1 and attachment specifications.

Connection type	Message type	
java.xml.soap.SOAPConnection	Point-to-point	Not for Message Provider, For Request-Response model.
Java.xml.soap.ProviderConnection	Message Provider	For One-way messaging.

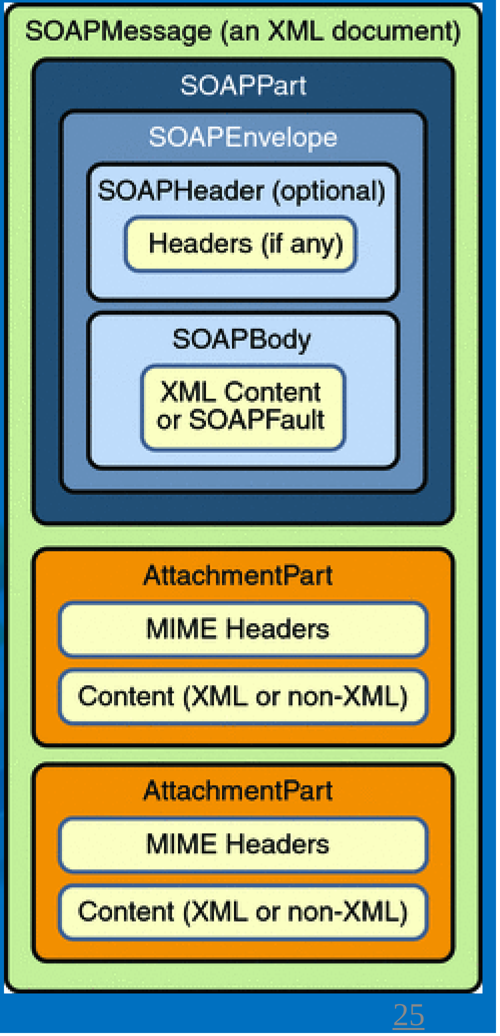
Message Types



SOAP with Attachments (SAAJ)

- Standard way to fabricate SOAP conformed XML documents.
- Java API to fabricate message.
- Messages conform to SOAP 1.1 and SOAP 1.2

Components	Description
SOAPMessage	SOAPPart exists but AttachmentPart is optional.
SOAPPart	SOAPEnvelope exists. It represents XML content only.
SOAPEnvelope	Empty SOAPHeader and SOAPBody
SOAPHeader	Info. About sending and receiving parties, intermediate destinations if any, message sequence, routing and delivery information.
SOAPBody	For XML information. May be optional if attachment exists.
AttchmentPart	Optional. If exists, one occurrence for an attachment,



JAX-WS Tools- wsimport

- Wsimport: For top down approach.
- WSDL -> Artifacts for invocation of services.
- Find command at JAVA_HOME/bin/wsimport.exe (wsimport.sh for unix).
- Creates following artifacts…
  - Service Endpoint Interface
  - Mapping of faults.
  - JAXB generated value types.

JAX-WS Tools- wsgen

- wsgen: For bottom-up approach.
- Service Implementation Class -> Jax-ws portable Artifacts for development and deployment of services.
- Find command at JAVA_HOME/bin/wsgen.exe (wsimport.sh for unix).

REST Web Service style

REST (REpresentational State Transfer)

- An architectural style of exposing web services on web;
- It is neither technology nor standard;
- A resource is a piece of information like book, order etc;
- Service is identified using URI;
- Client queries or update a resource through URI which influences state change in the resource i.e. REST;

SOAP Vs. REST

S.N.	SOAP	REST
01	Exposes services	Exposes data
02	It is standard	Yet in evolving phase. Not a standard
03	It is complex	Simple, fast and scalable
04	Allows exchange of XML data format	Allows exchange of any kind of data format including XML, JSON.
05	Encourages loose coupling	Couples tightly both ends.
06	Advertises themselves	Does not advertise
07	Reads can not be cached	Reads can be cached.
08	Preferred when provider changes data formats, API frequently. Prefer for complex API	For simple, stable API.

SOAP Communication Styles

The styles are defined in WSDL as SOAP binding.

Document:

Deals with XML Document as payload adhere to well defined contracted created using XML Schema definition.

The XML Schema definition specifies contract of XML Schema messages being exchanged in the form of Request and Response.

The Schema defines format and types for Request and Response.

RPC(Remote Procedure Call):

Focuses on XML representation of a method. Messages are not tied to XML Schema therefore can not be validated. RPC working in conjunction with SOAP encoding are called as RPC/encode otherwise RPC/literal. Not preferred because of inoperability issues.

RESTful Service

Web services built using principles of REST architecture are termed as RESTful web services.

1. Encapsulates data in simple data format like XML, JSON.
2. Simplifies development of services
3. Lightweight compared to SOAP.
4. Useful when simple data format is to be transmitted.
5. It exposes set of operations using standard HTTP methods like GET, POST, PUT.
6. Client invokes methods defined on resources using URI over HTTP protocol.

Principles of RESTful Service

- Addressable resources: Unique identity for each data chunk.
- Uniform constraint interface: Name activity with HTML-Method
- Representation oriented: Represents data format with MIME.
- Stateless communication: For better scalability

RESTful Service

URI for RESTful service:

http://localhost:9000/categoryservice/category/book

GET: Return list of all books from given categories
POST: Insert new book in given category.
DELETE: Remove a book from given category.

URI for RESTful service:

http://localhost:9000/categoryservice/category/book/001

GET: Return record for book ‘001’
PUT: Update record for book ‘001’ (Idempotent operation).
DELETE: Remove a book ‘001’.

JAX-RS specifications

- **POJO Centric**: JAX-RS API provides annotations, interfaces which can be applied on POJO to expose it as RESTful resource.
- **HTTP Centric**: Specifies clear mapping between HTTP protocol and JAX-RS API classes and methods. Specifies URL resolving algorithm to resource methods.
- **Format Independence**: Based on content type, RESTful mechanism should serve the request and create response.
- **Container Independence**: Specifies how RESTful services should be hosted on container including JEE container.
- **Client Side**: Does not talk about client side API

Developing RESTful web service

- Steps:
  1. Create Java data objects for Request and Response
  2. Configure binding for request and response objects.
  3. Create implementation and annotate it with JAX-RS.
  4. Test implementations.
  5. Deploy restful service in container.
  6. Write client code and fetch services.

Thanks